What is claimed is:

1. A method for balancing the load of a parallel processing system having a plurality of parallel processing elements arranged in a loop, wherein each processing element (PE_r) has a local number of tasks associated therewith, wherein r represents the number for a selected processing element, and wherein each of said processing elements is operable to communicate with a clockwise adjacent processing element and with an anti-clockwise adjacent processing element, the method comprising:

determining a total number of tasks present within said loop;

calculating a local mean number of tasks for each of said plurality of processing elements;

calculating a local deviation for each of said plurality of processing elements;

determining a running partial deviation sum for each of said plurality of processing elements;

determining a clockwise transfer parameter and an anti-clockwise transfer parameter for each of said plurality of processing elements; and

redistributing tasks among said plurality of processing elements in response to said clockwise transfer parameter and said anti-clockwise parameter for each of said plurality of processing elements.

2. The method of claim 1 wherein said determining a total number of tasks present within said loop, comprises:

transmitting said local number of tasks associated with each of said plurality of processing elements to each other of said plurality of processing elements within said loop;

receiving within each of said plurality of processing elements said number of local tasks associated with said each other of said plurality of processing elements; and

summing said number of local tasks associated with each of said plurality of processing elements with said number of local tasks associated with each other of said plurality of processing elements.

3. The method of claim 1 wherein said determining said total number of tasks present within said loop includes solving the equation $V = \sum_{i=0}^{i=N-1} v_i$, where N represents the number of said processing elements in said loop and v_i represents said local number of tasks associated with an i^{th} processing element in said loop.

4. The method of claim 1 wherein said calculating a local mean number of tasks within each of said plurality of processing elements includes solving the equation $M_r = Trunc((V + E_r)/N)$, where M_r is said local mean for said PE_r, N is the total number of said processing elements in said loop, and E_r is a number in the range of 0 to $(N-1)$ and wherein each processing element has a different E_r value.

5. The method of claim 4 wherein E_r controls said *Trunc* function such that said total number of tasks for said loop is equal to the sum of the local mean number of tasks for each of said plurality of processing elements in said loop.

6. The method of claim 4 wherein said local mean $M_r = Trunc((V + E_r)/N)$ for each local PE_r within said loop is equal to one of X and $(X+1)$.

7. The method of claim 1 wherein said calculating a local deviation within each of said plurality of processing elements comprises finding the difference between said local number of tasks for each of said plurality of processing elements and said local mean number for each of said plurality of processing elements.

8. The method of claim 1 wherein said determining a running partial deviation sum for each of said plurality of processing elements comprises:

transmitting said local deviation associated with each of said plurality of processing elements to another of said plurality of processing elements within said loop;

receiving within each of said plurality of processing elements said local deviation associated with at least one other of said plurality of processing elements; and

summing said local deviation associated with each of said plurality of processing elements with said local deviation associated with at least one other of said plurality of processing elements.

9. The method of claim 1 wherein said determining a running partial deviation sum for each of said plurality of processing elements comprises solving the equation $S_j = \sum_{i=0}^{i=j} D_i$, where S_j represents said running partial deviation sum, D_i represents the local deviation

associated with the i^{th} processing element, and $j \neq (N-1)$ where N is the number of processing elements on said loop.

10. The method of claim 1 wherein said determining a clockwise transfer parameter and an anti-clockwise transfer parameter within each of said processing elements comprises:

setting $T_a = (H_r + L_r) \div 2$; and

setting $T_c = D - T_a$ where H_r represents a highest extrema of said running partial deviation sum; L_r represents a lowest extrema of said running partial deviation sum, D represents the local deviation of a selected local processing element; and T_c represents said clockwise transfer parameter, and T_a represents said anti-clockwise transfer parameter.

11. A method for reassigning tasks associated with a selected processing element within a parallel processing system having a plurality of processing elements connected in a loop, each of said plurality of processing elements having a local number of tasks associated therewith, the method comprising:

determining the total number of tasks on said loop;

computing a local mean value for said selected processing element;

computing a local deviation for said selected processing element, said local deviation representative of the difference between said local number of tasks for said selected processing element and said local mean value for said selected processing element;

determining a running partial deviation sum for said selected processing element;

computing a number of tasks to transfer in a clockwise direction for said selected processing element;

computing a number of tasks to transfer in an anti-clockwise direction for said selected processing element; and

reassigning said tasks associated with said selected processing element relative to the said number of tasks to transfer in a clockwise direction and said number of task to transfer in an anti-clockwise direction.

12. The method of claim 11 wherein said determining the total number of tasks on said loop, comprises:

receiving within said selected processing element said number of local tasks associated with said each other of said plurality of processing elements; and

summing said number of local tasks associated with said selected processing element and said number of local tasks associated with each other of said plurality of processing elements.

13. The method of claim 11 wherein computing a local mean value for a selected processing element includes solving the equation $M_r = Trunc((V + E_r)/N)$, where M_r represents said local mean for a PE_r, N represents the total number of processing elements in said loop, and E_r is a number in the range of 0 to $(N-1)$.

14. The method of claim 13 wherein E_r controls said *Trunc* function such that said total number of tasks for said loop is equal to the sum of the local mean number of tasks for each of said plurality of processing elements in said loop and wherein each processing element has a different E_r value assigned.

15. The method of claim 11 wherein said determining a running partial deviation sum for said selected processing element comprises:

receiving within said selected processing element a local deviation associated with at least one other of said plurality of processing elements; and

summing said local deviation associated with said selected processing element and said local deviation associated with at least one other of said plurality of processing elements.

16. The method of claim 11 wherein said determining a running partial deviation sum for said selected processing element comprises solving the equation $S_j = \sum_{i=0}^{i=j} D_i$, where S_j represents said running partial deviation sum, D_i represents the local deviation associated with an i^{th} processing element, and $j \neq (N-1)$ where N is the number of said plurality of processing elements on said loop.

17. The method of claim 11 wherein said computing a local mean value, said computing a local deviation, said determining a running partial deviation sum, computing a number of tasks to transfer in a clockwise direction, computing a number of tasks to transfer in an anti-clockwise direction, and said reassigning tasks relative to the said number of task to transfer in a clockwise direction and said number of tasks to transfer in an anti-clockwise direction are completed simultaneously for each of said plurality of processing elements within said loop.

18. The method of claim 11 wherein said computing a number of tasks to transfer in a clockwise direction for said selected processing element includes evaluating at least one of a maximum extrema and a minimum extrema of said running partial deviation sum.

19. The method of claim 11 wherein said computing a number of tasks to transfer in an anti-clockwise direction for said selected processing element includes evaluating at least one of a maximum extrema and a minimum extrema of said running partial deviation sum.

20. A memory device carrying a set of instructions which, when executed, perform a method comprising:

determining a total number of tasks present within said loop;

calculating a local mean number of tasks for each of said plurality of processing elements;

calculating a local deviation for each of said plurality of processing elements;

determining a running partial deviation sum for each of said plurality of processing elements;

determining a clockwise transfer parameter and an anti-clockwise transfer parameter for each of said plurality of processing elements; and

redistributing tasks among said plurality of processing elements in response to said clockwise transfer parameter and said anti-clockwise parameter for each of said plurality of processing elements.